Attorney Docket No. 16869N-091900US
Client Ref. No. NT1274US

PATENT APPLICATION

Method and System for Job Management

Inventors: **Masanori HONDA**
Citizenship: Japan

Toshaiki HIRATA
Citizenship: Japan

Shigeru MIYAKI
Citizenship: Japan

Yasuke IZUMIDA
Citizenship: Japan

Takaki KURODA
Citizenship: Japan

ATsushi MURASE
Citizenship: Japan

Taro INOUE
Citizenship: Japan

Kenichi KIHARA
Citizenship: Japan

Assignee: Hitachi, Ltd.
6, Kanda Surugadai 4-chome
Chiyoda-ku, Tokyo, Japan
Incorporation: Japan

Entity: Large

TOWNSEND AND TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
(415) 576-0200

TITLE OF THE INVENTION

METHOD AND SYSYTEM FOR JOB MANAGEMENT

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to a method and system for job management within an information processing system, a program for executing job management, and a recording medium for storing such a program.

Description of the related Art

In an information processing system that comprises a computer serving as an information processing device, the jobs to be executed by the computer are generally set by writing a job control language with an editor or using a job setup tool. Some other systems use a GUI (Graphical User Interface) to perform job setup.

At the present time when the information technology is advanced, a professional, complicated technology is demanded for use in job setup. For job setup for an information processing system having a large-scale, complicated configuration, it is

particularly demanded that job setup be performed efficiency while considering how the CPU, memory, disk, backup device, and other resources are used. Meanwhile, for the definition of a job network that executes a stream of coordinated jobs, an advanced technology is required for executing the jobs while smoothly coordinating the jobs. Since increased workload is imposed on users, operators, and other job setup persons, a scheme for efficiently managing the jobs to be executed by a computer system is now called for.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a job management method, an information processing system, a program, and a recording medium.

Preferably, the present invention provides a job management method for an information processing system, which includes an information processing device. The job management method of the present invention comprises the steps of storing a stencil for a job definition statement and data prescribing a user interface for job definition statement setup; generating data for executing the process for generating a job definition statement based on the contents set by a user via the user interface in accordance

with the stencil for the job definition statement and the data prescribing the user interface for job definition statement setup; and generating the job definition statement by executing the process in accordance with the generated data.

The job definition statement is written, for instance, in a policy definition XML file, which is described later. The stencil for a job definition statement is written, for instance, in a policy template definition XML file, which is described later. The data for prescribing the user interface for job definition statement setup is, for instance, a wizard page, which is an element for policy template definition as described later. A stencil for a wizard page is written in a policy template definition XML file, which is described later. The user interface is, for instance, a setup guidance window (wizard window) that is generated by a policy wizard GUI, which is described later, in compliance with the contents of the wizard page for policy template definition. The user interface may be such a window or may be based on perception in any of the five senses including hearing.

In addition to a job definition statement stencil, the present invention provides a data which prescribes the user interface for job definition statement setup. With such a data , it is possible to efficiently generate a user interface (e.g.,

setup guidance window (wizard window)) for letting a user, operator, or other similar person generate a job definition statement. The present invention also makes it possible to offer a flexible user interface in accordance with the information processing system configuration, elements, user needs, applicable operations, and the like, thereby reducing the workload on users, operators, and other job definition setup persons.

The other features and advantages of the present invention will be apparent from the following detailed description and from the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates the hardware configuration of an information processing system according one embodiment of the present invention.

FIG. 2 illustrates the configuration and scheme of an information processing system according one embodiment of the present invention.

FIG. 3 shows elements that are unique to an information processing system according one embodiment of the present invention.

FIG. 4 shows a DTD sample of a policy definition XML file according one embodiment of the present invention.

FIG. 5 illustrates the structure of a policy template according one embodiment of the present invention.

FIG. 6 shows major elements of a policy template according one embodiment of the present invention.

FIG. 7 shows an example of a statement written in a policy stencil according one embodiment of the present invention.

FIG. 8 shows an example of a statement written in a policy stencil according one embodiment of the present invention.

FIG. 9 shows a DTD sample of a policy template definition XML file according one embodiment of the present invention.

FIG. 10 is a PAD illustrating a process that a policy template parser according one embodiment of the present invention performs in relation to an import function.

FIG. 11 is a PAD illustrating a process that a policy wizard GUI according to one embodiment of the present invention performs to generate a policy rule.

FIG. 12 is a PAD that illustrates a process concerning wizard page elements according to one embodiment of the present invention.

FIG. 13 shows a concrete example of an information processing system to which a job management system is applied in

accordance with one embodiment of the present invention.

FIG. 14 is a block diagram illustrating the configuration of a management server according to one embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

FIG. 1 illustrates the hardware configuration of an information processing system, which will be described as one embodiment of the present invention. The information processing system includes a computer 110, which comprises a CPU 111 and a memory 112, which includes a ROM and a RAM; a display or other display device 120; an input interface 130, which includes a keyboard and a mouse; and a hard disk drive, CD-ROM, or DVD-ROM, or other external storage device 140. The computer 110 may a mainframe computer, workstation, personal computer, or the like. A UNIX (registered trademark) or other operating system runs on the computer 110. Various application programs run on the operating system. On the operating system, a Web server also runs to offer a Web page on which a setup guidance window described later is written. The data stored in the external storage device 140 is managed on an individual file basis by the functionality of a file system that belongs to the operating system.

In the present embodiment, a job is a unit of processing to be performed by the computer 110 as viewed from a user of the computer 110. In the information processing system of the present embodiment, a job management system operates and provides an automatic job execution scheme. The job management system is implemented by a program that is executed by the computer 110. When used with a bank's on-line system, the job management system serves, for instance, as a batch processing system for performing transaction data totalization, analysis, data backup, and other operations. The job management system receives job setup instructions and conditions entered by a user, operator, or other similar person via the input interface 130, and executes a specified job under specified conditions. The conditions include conditions concerning a schedule for specifying the job execution date/time. As regards job network setup for executing a stream of a plurality of jobs, the conditions include those which are based on inter-job restrictions such as the succeeding job execution conditions appropriate for the status of a preceding job process execution (e.g., processing in progress or processing completed).

FIG. 2 illustrates the configuration and scheme of the job management system. The job management system provides a user interface 220, which allows a user, operator, or other similar

person to set a job definition statement (hereinafter referred to as a policy rule). The job management system stores a policy rule, which is set by the user interface, in the memory 112 of the computer 110 or in the external storage device 140 as a file in XML (Extensible Markup Language) format. An XML file in which a policy rule is written is hereinafter referred to as a policy definition XML file 210.

The above user interface 220 is implemented mainly by a function of a policy wizard GUI 230 shown in FIG. 2. In a repository 240, which is a database where various items of information about the job management system are registered and managed, a policy template definition 250, which includes the model data for a policy rule and the data stipulating the user interface 220, is registered.

In accordance with the policy template definition 250 registered in the repository 240, the policy wizard GUI 230 generates a Web page on which a series of setup guidance windows (wizard windows) organized to prompt the user to set up a job definition statement are written, as the above user interface 220, and then sequentially displays the generated setup guidance windows. The policy template definition 250 is the data that is used to execute the process for generating a job definition statement in accordance with the contents set by the user via the

user interface 220. The policy wizard GUI 230 generates a policy rule by incorporating the contents, which are set by the user in accordance with the setup guidance windows, into the model data for a policy rule.

The policy rule generated by the policy wizard GUI 230 is interpreted by a policy execution engine 260 shown in FIG. 2 and then registered in the repository 240 as policy definition information/execution time information (schedule information) 270. The policy execution engine 260 exercises job execution control in accordance with the policy definition information/execution time information 270 registered in the repository 240.

A policy control GUI 280, which is shown in FIG. 2, is used by a user, operator, or other similar person as the GUI for managing the repository 240 and policy execution engine 260. The user, operator, or other similar person can use the policy control GUI 280, for instance, to reference, delete, validate, or invalidate a policy rule. The policy control GUI 280 enables the user, operator, or other similar person to exercise post-management over a previously set policy rule.

The above-mentioned policy template definition 250 is registered in the repository 240 by a policy template parser 290 shown in FIG. 2. A policy template definition XML file 292, which

is shown in FIG. 2, is referenced when the policy template parser 290 registers the policy template definition 250 in the repository 240. In the policy template definition XML file 292, a stencil for a job definition statement and the data prescribing the user interface for job definition statement setup are written in XML format. The stencil can be used for a plurality of job definition statements and the data can be used for a plurality of policy template definitions in a modified embodiment.

The user, operator, or other similar person delivers an edited policy template definition XML file 292 to the policy template parser 290 when executing the policy template parser 290. This delivery is made, for instance, by specifying the name of the policy template definition XML file 292 as an argument for the command for executing the policy template parser 290. In accordance with the contents of the delivered policy template definition XML file 292, the policy template parser 290 generates a policy template definition 250 and registers it in the repository 240.

As described above, the policy template definition 250 can be generated and registered in the repository 240 with high efficiency by editing the policy template definition XML file 292. Further, the efficiency with which the policy template definition 250 is generated can be enhanced because the policy template

definition XML file 292, which contains a stencil for a job definition statement appropriate for the policy rule to be generated and the data prescribing the user interface for job definition statement setup, is available from the beginning. Therefore, the information processing system administrator or other similar person can efficiently generate a user interface (e.g., a wizard window or other setup guidance window) that permits the user, operator, or other similar person to generate a job definition statement.

When the above stencils are used for editing the policy template definition XML file 292 as described above, the workload on the job management system administrator or other similar person can be reduced so that the policy template definition 250 can be generated with high efficiency. This advantage also makes it possible to offer various types of user interfaces in a manner flexible enough to match the information processing system configuration, elements, user needs, applicable operations, and other conditions, thereby reducing the workload on users, operators, and other job definition statement setup persons.

Policy rule:

As described above, the policy rule is managed as a policy definition XML file 210, which is in XML format. When the policy rule is managed as an XML-formatted file in a manner described

above, it is possible, for instance, to easily exchange the policy rule with a remote information processing system. As a result, existing assets possessed by a remote information processing system can be effectively used while supplying assets to such a remote information processing system.

Further, when XML is used as a descriptive syntax for writing the policy rule, it is possible to (1) easily grasp the relationship to a window by noting the name of a tag, (2) assign a unique name to the contents of information, (3) freely determine the correlation between a tag and element, and make use of the other features proper to XML. In addition, future expandability can be provided by incorporating newly defined XML specifications.

The descriptive syntax for the policy definition XML file 210 basically conforms to that of XML. The job management system of the present embodiment, however, is capable of not only complying with the basic descriptive syntax of XML but also interpreting various elements and executing a corresponding process. Major elements unique to the information processing system are shown in FIG. 3. In this figure, the policy is a root element for the policy definition XML file 210. The policy group is an element for grouping a policy group and a root policy rule described later. The root policy rule is an element for grouping

a policy condition and a policy action, which are described later, as well as the above policy rule.

The policy wait event condition is an element that is written when an event is to be awaited. This element is used for writing an event-driven policy. The policy schedule condition is an element that is written when a specified time is to be awaited. This element is written for executing a policy command action, which is described later, at a predetermined time. The policy command result condition is an element that is written when the end of a policy command action is to be awaited. If the policy command action execution result is a specified value, the policy command action serving as a sibling element will be executed.

The policy rule validity period is an element that is written to define the validity period of a root policy rule. The compound policy action is an element for grouping policy actions. The policy command action is a command that is executed when a policy wait event condition, policy schedule condition, policy command result condition, or other condition defined as a sibling element is established.

In the policy definition XML file 210, a global environment variable can be defined. The global environment variable can be commonly referenced by all policy command actions that are written as lower-level elements for a certain root policy rule. In an

information processing system containing a plurality of computers, the computer executing a policy command action is not always the same. The global environment variable is useful if the variable to be commonly referenced by all policy command actions is to be provided in situations where a job targeted for a plurality of computers is set. Each policy command action can set an initial value for the global environment variable. The initial value for the global environment variable can also be set for a policy group and a root policy rule.

In the policy definition XML file 210, the elements described above are written in compliance with the XML syntax. In other words, the policy wizard GUI generates the policy definition XML file 210 by incorporating the contents, which are set by the user, operator, or other similar person in relation to the user interface 220, into the model data for the above policy rule contained in the policy template definition in accordance with the policy template definition registered in the repository 240. FIG. 4 shows a typical DTD (Document Type Definition) section as an example of the generated policy definition XML file 210.

Policy template definition:

As described above, the policy template definition 250 is a description of the model data for a policy rule and the data

prescribing the above user interface 220. The policy template definition 250 contains the above setup guidance windows, which the policy wizard GUI 230 displays for policy rule setup, as well as the information serving as a model for the policy definition XML file 210 that is to be generated.

FIG. 5 illustrates the structure of a policy template definition 250. FIG. 6 shows major elements of the policy template definition 50. In FIG. 5 or FIG. 6, the policy template set is a root element for all elements. The policy template group is an element for grouping policy templates. The policy template group can be written in a nested form. The policy template group plays a role similar to that of the file system's folder (directory). The policy template is an element for formulating various definitions for generating a policy rule.

The policy stencil is an element that defines the model of a policy rule (policy definition XML file). A statement can be written on the policy stencil. For example, it is possible to write a statement for exercising control so that the description between the start statement and end statement will be written into the policy definition XML file 210 as a valid description if the expression is true and will not be written if the expression is false. An example of such a statement is shown in FIG. 7 (the statement in this figure is a command named "@Validf"). It is

also possible, for instance, to sequentially substitute list elements into a variable and write a statement for exercising control so as to repeatedly write the description between the start and end statements concerning each element into the policy definition XML file 210. An example of such a statement is shown in FIG. 8 (the statement in this figure is a command named "@ForEach").

The wizard page is an element for defining a setup guidance window (wizard window). Each wizard page corresponds to one page of a wizard window (a single Web page). On the wizard page, it is possible to write control data for specifying whether or not to display user-definable options in the wizard window. It is also possible to write control data for specifying whether or not to display a setup guidance window that can open subsequently to a preceding setup guidance window depending on a user response to the preceding setup guidance window.

For one wizard page, one or more parameter setup controls can be defined. Each parameter setup control is an element for finalizing a policy template variable (value substitution). One parameter setup control finalizes one policy template variable. Three different parameter setup controls are available: Text Line, Text Area, and Select Item. The Text Line control is used for specifying one character string line. The Text Area control is

used for specifying a plurality of character string lines. The Select Item control is used for displaying a plurality of items and selecting a specific item from them. The policy template variable is a variable that is declared for a policy template. The method for finalizing its value is defined by a parameter setup control within a wizard page, and the value is referenced within a policy stencil or wizard page. The resource type is an element for defining which resource can be associated with a policy template.

Policy template definition XML file, etc.:

As described above, the policy template definition XML file 292 is a file in which the data for generating a policy template definition 250 is written in XML format. The policy template definition XML file 292 contains an XML statement, which is in the form of a policy stencil to serve as a stencil for the above policy definition XML file 210. The data prescribing the above wizard page within the above policy template definition 250 is written in XML format together with the stencil for a job definition statement. FIG. 9 shows a typical DTD section as an example of the policy template definition XML file 292.

The policy template parser 290 interprets an entered policy template definition XML file 292 in order to register a policy template definition 250 conforming to the policy template

definition XML file 292 in the repository 240. For example, the policy template parser 290 is implemented within the information processing system as a Java (registered trademark) class. Further, the policy template parser 290 offers a CLI (Command Line Interface) function as needed so that it can be started up from the command line.

The import function of the policy template parser 290 will now be described. The policy template parser 290 exercises its import function for the purpose of registering a policy template definition 250 in the repository 240 in accordance with the policy template definition XML file 292. When, for instance, the input interface 130 issues an instruction for executing the import function, the policy template parser 290 reads and interprets the policy template definition XML file 292. When there is no problem with the result of interpretation, the policy template parser 290 registers in the repository 240 a policy template definition 250 that is generated in accordance with the policy template definition XML file 292.

FIG. 10 is a PAD (Problem Analysis Diagram) illustrating a process that the policy template parser 290 performs in relation to the import function. When an instruction for executing the import function, the policy template parser 290 first clears a variable, sets an initial value, and performs other initial

processing operations (step S1011). The policy template parser 290 also analyzes an argument that is incidentally specified for the instruction for import function execution (step S1012). Next, the policy template parser 290 views the contents of the repository 240 (step S1013) to check for a problem such as an inconsistency between the contents of the repository 240 and argument (step S1014). If any problem is found, the policy template parser 290 aborts the current process and then performs exception handling (step S1015). If no problem is found, on the other hand, the policy template parser 290 analyzes the contents of the policy template definition XML file 292 corresponding to the file name specified as an argument, and registers in the repository 240 a policy template definition 250 that corresponds to the policy template definition XML file 292 (step S1016). The process concerning the import function is then completed (step S1017).

The policy template parser 290 for the job management system of the present embodiment has various other functions in addition to the above import function, including an export function for generating a policy template definition XML file 292 in accordance with a policy template definition 250 registered in the repository 240 and a function for deleting a policy template definition 250 registered in the repository 240.

Policy rule setup:

The user, operator, or other similar person creates a policy rule in accordance with setup guidance windows displayed by the policy wizard GUI 230. FIG. 11 is a PAD illustrating a process that the policy wizard GUI 230 performs to generate a policy rule. The policy wizard GUI 230 first clears a variable, sets an initial value, and performs other initial processing operations (S1111), then analyzes an argument that is incidentally specified for the instruction for import function execution (S1112). Next, the policy wizard GUI 230 views the repository 240 (S1113) to check for a problem such as an inconsistency between the contents of the repository 240 and argument (S1114). If any problem is found, the policy wizard GUI 230 aborts the current process and then performs exception handling (S1115). Next, the policy wizard GUI 230 reads a policy template element attribute, policy template variable element attribute, and policy stencil element value from the repository 240 (S1116-S1118). Subsequently, the policy wizard GUI 230 initiates a process for all wizard page elements contained in the policy template definition 250 (S1119).

FIG. 12 is a PAD that illustrates a process concerning wizard page elements. The policy wizard GUI 230 first reads the attribute of a wizard element from the repository 240 (S1211). In this instance, the policy wizard GUI 230 also evaluates the policy template variable. Next, the policy wizard GUI 230 checks

whether the page should be skipped (skip rule evaluation in the figure) (S1212). If the page should be skipped (S1213), the policy wizard GUI 230 terminates the wizard page element process (S1214). If there is the next wizard page element to be processed, the policy wizard GUI 230 starts the process for it. If there is no more wizard element to be processed, the policy wizard GUI 230 proceeds to perform processing step S1220.

In processing step S1215, the policy wizard GUI 230 starts a process for the parameter setup controls contained in an element of the wizard page. The policy wizard GUI 230 first reads the attribute of a parameter setup control from the repository 240. In this case, the policy wizard GUI 230 also evaluates the policy template variable. The parameter setup control mentioned here is "TextLine," "TextArea," or "SelectItem." If there is a child element for the parameter setup control, the policy wizard GUI 230 reads and evaluates its attribute. In this instance, the policy wizard GUI 230 also evaluates the policy template variable (S1217).

In the next processing step (S1218), a setup guidance window (wizard window) appears on the display device 120 in accordance with the read attribute and other data. The user performs setup in relation to this setup guidance window. Upon completion of setup, the user clicks the Next button or performs some other

operating step to switch to the next window (S1219). Although the description of the present embodiment assumes a relatively simple user operation, the contents of the setup guidance windows to be displayed and their display method are not limited to those described herein.

When all the wizard page elements are completely processed (S1200) as a result of processing depicted in FIG. 11 (S1119), the policy wizard GUI 230 opens a parameter confirmation window (S1120). The user, operator, or other similar person confirms the parameters, sets a policy rule, and performs other setup operations from the parameter confirmation window, which is opened by a setup guidance window, and clicks the Finish button if there is no problem (S1121). Next, the policy wizard GUI 230 substitutes the value of the policy template variable into the policy stencil (S1122). Further, the policy wizard GUI 230 evaluates a policy stencil control statement to complete the policy rule (S1123).

The completed policy rule is output as a policy definition XML file 210 and delivered to the policy execution engine 260 (S1124). This delivery is made by the functionality of a policy parser, which is not shown. The policy parser interprets the completed policy definition XML file 210 to generate a policy rule instance and then registers it in the policy execution engine 260.

Upon completion of the above process, the policy wizard GUI 230 opens a policy rule generation completion window (S1125) and terminates a policy rule (policy definition XML file) generation process (S1126).

If a plurality of policy template definitions 250 are registered in the repository 240, the process illustrated in FIG. 11 is performed for each policy template definition 250. In situations where a plurality of policy template definitions 250 are registered in the repository 240, setup guidance windows appropriate for individual policy template definitions 250 are offered.

Preferable application:

A preferable application of an information processing system in which the above job management system operates will now be described. The FIG. 13 illustrates an information processing system that functions as a typical storage system, which is operated, for instance, at a corporation's system center or a privately/publicly managed data center. In this storage system, a program for implementing the job management system runs on a management server 1320. The management server 1320 is a computer for servicing, operating, managing, or otherwise handling the storage system. The management server 1320 is used, for instance, to perform setup for making a storage device 1300 available to

an operation server 1310, set up a SAN, set the operation server 1310 to execute a batch process, and perform a process for allowing a tape device 1390 to store a data backup that is stored in the storage device 1300.

When a job to be performed by various devices within the information processing system in relation to the job management system running on the management server 1320 is defined as a policy rule by the operator or other similar person, the management server 1320 executes a command for controlling the various devices so that the various devices perform processes in compliance with the policy rule. If, for instance, the operator or other similar person sets a policy rule for making a backup at a preselected time, the management server 1320 executes a command for controlling a logical volume 1360, which will be described later, as well as the tape device 1390 so as to ensure that a process will be performed in compliance with the policy rule.

The management server (information processing device) 1320 is connected to the operation server 1310, FC-SW1 1350, the storage device 1300, and the tape device 1390 via a management LAN 1341.

FIG. 14 illustrates the configuration of the management server 1320. The management server 1320 comprises a CPU 1410, a memory 1411, a LAN interface 1412, a recording media reader 1420, an input device 1425, an output device 1426, and a storage device

1430. The recording media reader 1420 is a device for reading a program or data that is recorded on a recording medium 1421. For example, a program 1450 for implementing the above job management system, which is recorded on the recording medium 1421, can read from the recording medium 1421 by the recording media reader 1420 and stored in the memory 1411 or the storage device 1430. As the recording medium 1421, a flexible disk, CD-ROM, DVD-ROM, semiconductor memory, or other similar medium can be used.

As the storage device 1430, a hard disk drive, flexible disk drive, semiconductor memory, or other similar device can be used. The input device 1425 is used, for instance, when the user, operator, or other similar person enters data into the management server 1320. As the input device 1425, a keyboard, mouse, or other similar device can be used. As the output device 1426, a display, printer, or other similar device can be used. The LAN interface 1412 is a device for communicating with the operation server 1310, FC-SW1 1350, the storage device 1300, or the tape device 1390.

An operation client 1315 is connected to the operation server 1310 via a backbone LAN (Local Area Network) 1340. The operation server 1310 and the operation client 1315 are both computers that are equipped, for instance, with a CPU (Central Processing Unit), a memory, and an input/output device.

Therefore, the operation client 1315 can receive various services that are offered by the operation server 1310.

The services offered by the operation server 1310 include, for instance, on-line services such as a bank's automated teller service and Internet homepage browsing service, and a batch processing service for technological experiment simulations.

The operation server 1310 is connected to the storage device 1300 and the tape device (backup device) 1390 via FC-SW1 (Fibre Channel Switch 1, network device) 1350. FC-SW1 1350 is a switch for establishing a SAN (Storage Area Network) connection among the operation server 1310, the storage device 1300, and the tape device 1390.

The storage device 1300 supplies a storage resource as needed when the operation server 1310 offers an information processing service to the operation client 1315. For example, a disk array device can be used as the storage device 1300. The storage resource is supplied as a logical volume 1360. The logical volume 1360 is a storage area that is logically set within a physical storage region, which is provided by a disk drive incorporated in the storage device 1300. As the disk drive, a hard disk drive, flexible disk drive, semiconductor memory, or other similar device can be used. The disk drive may be managed as a RAID (Redundant Array of Inexpensive Disks) managed.

The present invention has been described with reference to a preferred embodiment. The present embodiment is considered in all respects to be illustrative, and not restrictive. Therefore, the present invention is not limited to a particular embodiment, but extends to various modifications that nevertheless fall within the scope of the appended claims or the equivalence thereof.